

IN THE U.S. PATENT AND TRADEMARK OFFICE  
Patent Application Transmittal Letter

COMMISSIONER FOR PATENTS  
Washington, D.C. 20231

Sir:

Transmitted herewith for filing under 37 CFR 1.53(b) is a(n): ☒ Utility ☐ Design

☒ original patent application,

☐ continuation-in-part application

INVENTOR(S): Michael L. Ziegler, Carol L. Thompson

TITLE: Method and Apparatus for Resuming Execution of a Failed Computer Program

Enclosed are:

☒ The Declaration and Power of Attorney. ☒ signed ☐ unsigned or partially signed

☒ 4 sheets of drawings (one set) ☐ Associate Power of Attorney

☐ Form PTO-1449 ☐ Information Disclosure Statement and Form PTO-1449

☐ Priority document(s) ☐ (Other) (fee \$ )

CLAIMS AS FILED BY OTHER THAN A SMALL ENTITY				
(1) FOR	(2) NUMBER FILED	(3) NUMBER EXTRA	(4) RATE	(5) TOTALS
TOTAL CLAIMS	11 — 20	0	X \$18	\$ 0
INDEPENDENT CLAIMS	5 — 3	2	X \$80	\$ 160
ANY MULTIPLE DEPENDENT CLAIMS	0		\$270	\$ 0
BASIC FEE: Design (\$320.00 ); Utility (\$710.00 )				\$ 710
TOTAL FILING FEE				\$ 870
OTHER FEES				\$
TOTAL CHARGES TO DEPOSIT ACCOUNT				\$ 870

Charge \$ 870 to Deposit Account 08-2025. At any time during the pendency of this application, please charge any fees required or credit any over payment to Deposit Account 08-2025 pursuant to 37 CFR 1.25. Additionally please charge any fees to Deposit Account 08-2025 under 37 CFR 1.16, 1.17, 1.19, 1.20 and 1.21. A duplicate copy of this sheet is enclosed.

"Express Mail" label no. EL571600602US

Date of Deposit Nov. 28, 2000

I hereby certify that this is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to: Commissioner for Patents, Washington, D.C. 20231.

By Keri J. Kuhlmann  
Typed Name: Keri J. Kuhlmann

Respectfully submitted,

Michael L. Ziegler, Carol L. Thompson

By

LeRoy D. Maunu

Attorney/Agent for Applicant(s)

Reg. No. 35,274

Date: Nov. 28, 2000

Telephone No.: (651) 686-6633

1 METHOD AND APPARATUS FOR  
2 RESUMING EXECUTION OF A FAILED COMPUTER PROGRAM  
3

4 FIELD OF THE INVENTION

5 The present invention generally relates to recovery  
6 from errors in computer programs, and more particularly to  
7 using alternative code to recover execution of a failed  
8 computer program.

9  
10 BACKGROUND

11 Certain types of software errors are fatal to program  
12 execution. For example, a reference to a memory address  
13 that is beyond the address domain of a program will likely  
14 result in a fatal error. Certain timing or other  
15 transient conditions may also trigger fatal errors. While  
16 certain errors may be within the control of the software  
17 developer, the developer may be unable to guard against  
18 certain other errors in developing the software.

19 Program errors that are transient or timing-related  
20 are problematic to both the program user and to the  
21 program developer. From the user's point of view, failure  
22 of the program not only interrupts the task at hand, but  
23 may also result in lost work product. Reporting such  
24 transient errors is also difficult since the conditions  
25 required to reproduce the problem are likely to be  
26 unknown.

27 From the developer's point of view, much time may be  
28 spent trying to find the root cause of a problem where the  
29 root cause is external to the program. Furthermore,  
30 tracing the root cause of the problem may be difficult and  
31 time-consuming if there is there is scant information  
32 available for reproducing the problem.

33 A method and apparatus that address the  
34 aforementioned problems, as well as other related  
35 problems, are therefore desirable.

1

2 SUMMARY OF THE INVENTION

3 In various embodiments, a computer implemented method  
4 is provided for recovery from fatal program errors. A  
5 program is compiled using two compilers to generate first  
6 and second sets of object code. Checkpoints are  
7 identified in the program, and checkpoint code is  
8 generated for execution at the checkpoints. If execution  
9 of the first set of object code fails, checkpoint data is  
10 recovered and execution of the program is resumed using  
11 either the first or second set of object code. In one  
12 embodiment, the first set of object code is re-executed  
13 before trying the second set of object code. In another  
14 embodiment, the second set of object code automatically  
15 executed upon failure of the first set of object code.

16 It will be appreciated that various other embodiments  
17 are set forth in the Detailed Description and Claims which  
18 follow.

19

20 BRIEF DESCRIPTION OF THE DRAWINGS

21 Various aspects and advantages of the invention will  
22 become apparent upon review of the following detailed  
23 description and upon reference to the drawings in which:

24 FIG. 1 is a flowchart of a an example process for  
25 compiling a program in accordance with one embodiment of  
26 the invention;

27 FIG. 2 is a block diagram that illustrates an example  
28 data structure used in the management of checkpoints;

29 FIG. 3 is a block diagram that illustrates the code  
30 that results from compilation of a program in accordance  
31 with one embodiment of the invention; and

32 FIG. 4 is a flowchart of an example process for error  
33 recovery.

34

1 DETAILED DESCRIPTION

2 In various embodiments, the present invention  
3 provides a method and apparatus for generating alternative  
4 code that supports recovery from a fatal program error.

5 FIG. 1 is a flowchart of a an example process for  
6 compiling a program in accordance with one embodiment of  
7 the invention. The process generally entails generating  
8 two sets of object code segments using two different  
9 compilers or code generators. The second set of object  
10 code segments are available for execution in the event  
11 that the a fatal program error is encountered in executing  
12 the first set of segments. The segments of object code  
13 are delineated by checkpoints that are identified by the  
14 compiler.

15 At steps 202 and 204, the program source code is  
16 compiled using two different compilers, and two sets of  
17 object code that are functionally equivalent are created  
18 from the compilation. For example, the compilers may be  
19 different versions of the same compiler or compilers from  
20 different vendors. For making the two sets of object code  
21 interchangeable, checkpoints are identified and checkpoint  
22 code is generated for the two sets of object code at step  
23 206. Known computer platforms allow multiple compilers to  
24 coexist and also allow inter-operation at the  
25 function/procedure call level.

26 In one embodiment, the checkpoints are identified and  
27 the code generated as described in the co-pending patent  
28 application entitled, "Compiler-based Checkpointing for  
29 Support of Error Recovery" by Ziegler et al. and filed on  
30 October 31, 2000, which has patent/application number  
31 \*\*\*\*\* and attorney docket number 10001159, is commonly  
32 assigned to the assignee of the present invention, and the  
33 contents of which are incorporated herein by reference.  
34 Each segment of code is delineated by a checkpoint, as  
35 determined by the compiler. The checkpoints are points in

1 the code where the state of the program is stored so that  
2 execution can be recovered at the point in the program  
3 following the checkpoint. For example, convenient places  
4 for checkpoints are procedure boundaries.

5 At step 208, the code that performs the checkpointing  
6 is generated and combined with the intermediate code that  
7 was generated from the user's source code. Along with the  
8 checkpointing code, a data structure is created for  
9 storage of the checkpoint data at step 208.

10  
11 FIG. 2 is a block diagram that illustrates an example  
12 data structure used in the management of checkpoints in  
13 accordance with one embodiment of the invention. To save  
14 storage space, two checkpoint data sets 260a and 260b are  
15 maintained.

16 Checkpoint data are alternately stored in checkpoint  
17 data sets 260a and 260b for consecutive checkpoints. For  
18 example, at time t1, checkpoint data set 260a references  
19 checkpoint 262 and checkpoint data set 260b references  
20 checkpoint 264. At time t2 after program execution  
21 completes checkpoint 266, checkpoint data set 260a  
22 references checkpoint 266, and checkpoint data set 260b  
23 references checkpoint 264.

24 Timestamps or commit flags may be used in alternative  
25 embodiments to indicate which of the checkpoint data sets  
26 is to be used in recovery. The timestamp scheme involves  
27 writing a timestamp to a checkpoint data set when the  
28 storage of state information in the checkpoint data set is  
29 complete. Thus, the later of the two timestamps indicates  
30 which of checkpoint data sets 260a or 260b is to be used  
31 in recovery. The commit flag scheme involves a flag that  
32 indicates which of checkpoint data sets 260a or 206b is to  
33 be used in recovery.

34

1        FIG. 3 is a block diagram that illustrates the code  
2 that results from compilation of a program in accordance  
3 with one embodiment of the invention. One purpose for  
4 compiling code in the manner taught herein is to enable  
5 recovery from fatal program errors. Block 102 represents  
6 program source code that is to be compiled and is  
7 comprised of  $n$  segments of source code. Checkpoints are  
8 used to delineate the multiple segments. A checkpoint is  
9 a location in the code at which execution can recommence  
10 should the program encounter a fatal error. At each  
11 checkpoint, the state of data elements used by the program  
12 are stored so that in the event of program failure the  
13 state information can be recovered and execution resumed  
14 immediately after the checkpoint from which the state was  
15 recovered. In various embodiments the checkpoints can be  
16 user-programmed or identified by the compiler using  
17 recognized techniques.

18        Compilation of the program source code results in  
19 program object code 104 that includes two sets of object  
20 segments, object segments 1- $n$  and object segments 1- $n'$ .  
21 The object segments in each set correspond to the source  
22 segments of program source code 102.

23        Each set of object segments is code that is generated  
24 in compiling the source code with different compilers or  
25 code generators. In other words, object segments 1- $n$  are  
26 generated by a first compiler, and object segments 1'- $n'$   
27 are generated by a different compiler. If the program  
28 fails during execution of segment  $i$ , for example, then the  
29 state of the checkpoint data can be recovered from  
30 checkpoint that precedes segment  $i$  and execution can  
31 resume at segment  $i'$ .

32  
33        FIG. 4 is a flowchart of an example process for error  
34 recovery in accordance with one embodiment of the present  
35 invention. The process generally entails recovering from

1 a fatal program error by restoring checkpoint data and  
2 resuming execution of the program. In resuming execution,  
3 the code of the first set of segments is retried. If the  
4 failure is repeated, execution of the program is resumed  
5 using the alternative set of segments.

6 The process begins when a program error has been  
7 detected by the operating system, for example. At step  
8 304, checkpoint data is restored, and at step 306 the  
9 program counter is reset to the selected checkpoint. The  
10 segment of object code from the first set of segments is  
11 re-executed at step 308. If the program executes the  
12 segment without error, decision step 310 and step 312  
13 illustrate that the program continues with execution of  
14 the segments from the first set.

15 If an error recurs in executing the segment of code  
16 from the first set, control is directed to decision step  
17 314, which determines whether the alternative code should  
18 be tried. In one embodiment, the first set of segments of  
19 code may be re-executed a selected number of times before  
20 trying execution of the alternative code. Control is  
21 directed to step 316 when the decision is made to execute  
22 the alternative code.

23 At step 316, checkpoint data is restored, and at step  
24 318, the address of the segment of object code from the  
25 second set is selected for execution. That is, the  
26 program counter is loaded with a program address of a  
27 segment in the second set. At step 320, the alternative  
28 segment of object code is executed, and execution of the  
29 segments of code from the second set continues at step  
30 322.

31 Returning now to decision step 314, if the decision  
32 is made to not execute the alternative code, control is  
33 directed to decision step 324. Decision step 324 tests  
34 whether execution of the segment from the first set of  
35 segments should be attempted again. If so, control is

1 returned to step 304 to restore the checkpoint data and  
2 try again. Otherwise, the program is exited with an  
3 error.

4 The present invention is believed to be applicable to  
5 compilers for a variety of programming languages. Other  
6 aspects and embodiments of the present invention will be  
7 apparent to those skilled in the art from consideration of  
8 the specification and practice of the invention disclosed  
9 herein. It is intended that the specification and  
10 illustrated embodiments be considered as examples only,  
11 with a true scope and spirit of the invention being  
12 indicated by the following claims.

13



1 **CLAIMS**

2 What is claimed is:

3 1. A computer-implemented method for software error  
4 recovery, comprising:

5 compiling program source code into a first set of  
6 object code with a first compiler;

7 compiling the program source code into a second set  
8 of object code with a second compiler;

9 identifying checkpoints in the first and second sets  
10 of object code, each checkpoint in the first set of object  
11 code corresponding to a checkpoint in the second set of  
12 object code;

13 associating sets of data objects with the  
14 checkpoints;

15 automatically generating executable checkpoint code  
16 for execution at the checkpoints, the checkpoint code  
17 configured to store state information of the associated  
18 data objects for recovery if execution of the program is  
19 interrupted;

20 executing the first set of object code;

21 storing the state information in executing the  
22 checkpoint code; and

23 upon detecting an error in execution of the first set  
24 of object code, resuming execution of the program using  
25 the second set of object code.

26

27 2. The method of claim 1, further comprising:

28 upon detecting an error in execution of the first set  
29 of object code, initially re-executing the first set of  
30 object code; and

31 resuming execution using the second set of object  
32 code if the first set of object code fails in re-  
33 execution.

34

1 3. The method of claim 2, further comprising re-  
2 executing the first set of object code a selected number  
3 of times before resuming execution using the second set of  
4 object code.

5  
6 4. The method of claim 3, further comprising ceasing  
7 resumption of execution of the first and second sets of  
8 object code if an error is detected in executing both sets  
9 of object code.

10  
11 5. A computer-implemented method for software error  
12 recovery, comprising:

13 compiling program source code into a first set of  
14 object code with a first compiler;

15 compiling the program source code into a second set  
16 of object code with a second compiler;

17 identifying checkpoints in the first and second sets  
18 of object code, each checkpoint in the first set of object  
19 code corresponding to a checkpoint in the second set of  
20 object code;

21 associating sets of data objects with the  
22 checkpoints;

23 automatically generating executable checkpoint code  
24 for execution at the checkpoints, the checkpoint code  
25 configured to store state information of the associated  
26 data objects for recovery if execution of the program is  
27 interrupted;

28 executing the first set of object code;

29 storing the state information in executing the  
30 checkpoint code; and

31 upon detecting an error in execution of the first set  
32 of object code, selecting between the first set of object  
33 code and the second set of object code in resuming  
34 execution of the program.

35

1 6. The method of claim 5, further comprising:  
2 upon detecting an error in execution of the first set  
3 of object code, initially re-executing the first set of  
4 object code; and  
5 resuming execution using the second set of object  
6 code if the first set of object code fails in re-  
7 execution.

8  
9 7. The method of claim 6, further comprising re-  
10 executing the first set of object code a selected number  
11 of times before resuming execution using the second set of  
12 object code.

13  
14 8. The method of claim 7, further comprising ceasing  
15 resumption of execution of the first and second sets of  
16 object code if an error is detected in executing both sets  
17 of object code.

18  
19 9. An apparatus for software error recovery, comprising:  
20 means for compiling program source code into a first  
21 set of object code with a first compiler;  
22 means for compiling the program source code into a  
23 second set of object code with a second compiler;  
24 means for identifying checkpoints in the first and  
25 second sets of object code, each checkpoint in the first  
26 set of object code corresponding to a checkpoint in the  
27 second set of object code;  
28 means for associating sets of data objects with the  
29 checkpoints; and  
30 means for automatically generating executable  
31 checkpoint code for execution at the checkpoints, the  
32 checkpoint code configured to store state information of  
33 the associated data objects for recovery if execution of  
34 the program is interrupted;  
35 means for executing the first set of object code;

1 means for storing the state information in executing  
2 the checkpoint code; and  
3 means for resuming execution of the program using the  
4 second set of object code upon detecting an error in  
5 execution of the first set of object code.

6  
7 10. A computer-implemented method for software error  
8 recovery, comprising:

9 means for compiling program source code into a first  
10 set of object code with a first compiler;

11 means for compiling the program source code into a  
12 second set of object code with a second compiler;

13 means for identifying checkpoints in the first and  
14 second sets of object code, each checkpoint in the first  
15 set of object code corresponding to a checkpoint in the  
16 second set of object code;

17 means for associating sets of data objects with the  
18 checkpoints;

19 means for automatically generating executable  
20 checkpoint code for execution at the checkpoints, the  
21 checkpoint code configured to store state information of  
22 the associated data objects for recovery if execution of  
23 the program is interrupted;

24 means for executing the first set of object code;

25 means for storing the state information in executing  
26 the checkpoint code; and

27 selecting between the first set of object code and  
28 the second set of object code in resuming execution of the  
29 program upon detecting an error in execution of the first  
30 set of object code.

31

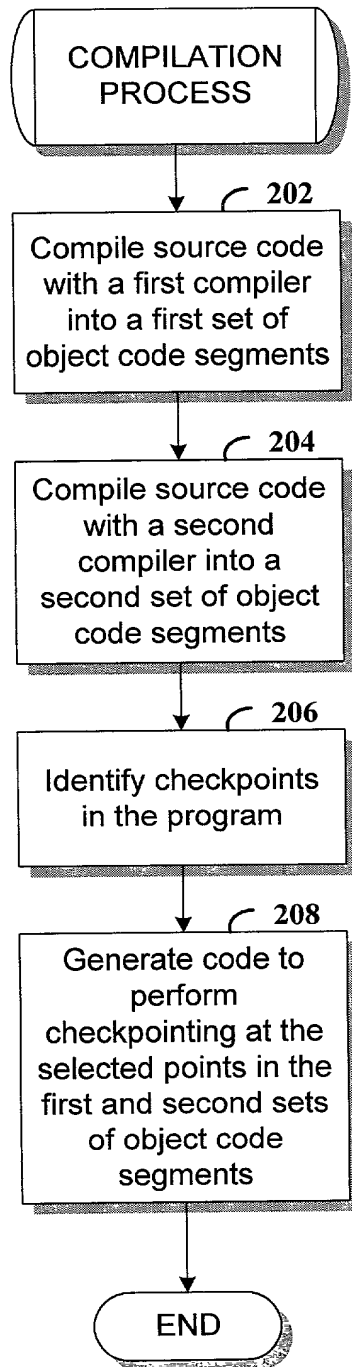
32 11. A computer program product configured for causing a  
33 computer to perform the steps comprising:

34 compiling program source code into a first set of  
35 object code with a first compiler;

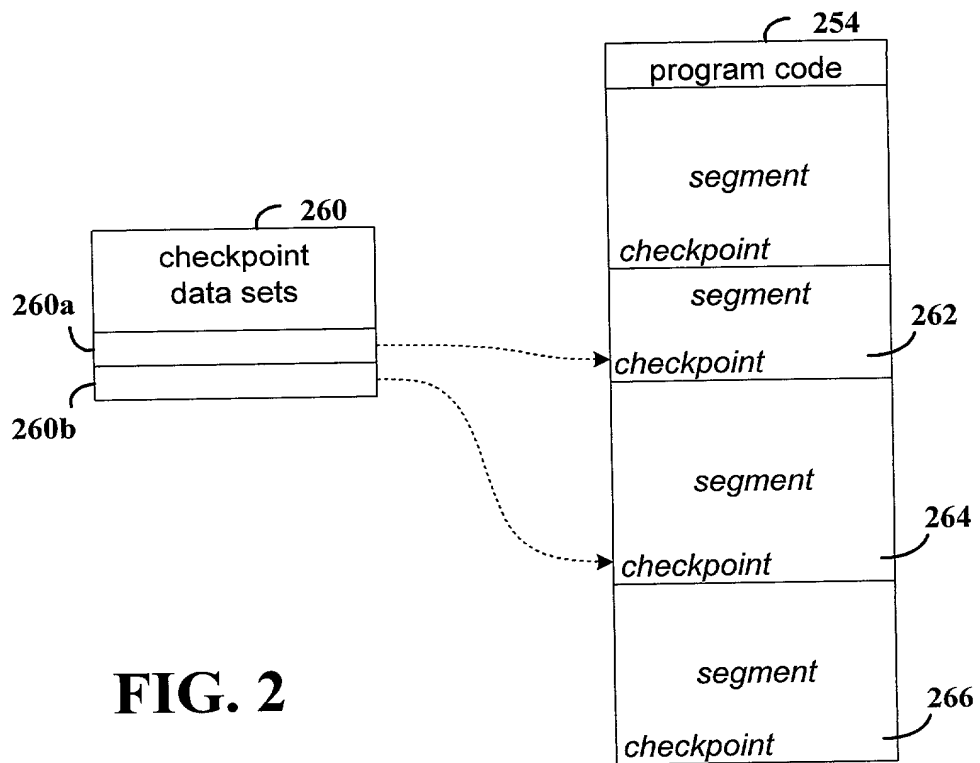
1        compiling the program source code into a second set  
2 of object code with a second compiler;  
3        identifying checkpoints in the first and second sets  
4 of object code, each checkpoint in the first set of object  
5 code corresponding to a checkpoint in the second set of  
6 object code;  
7        associating sets of data objects with the  
8 checkpoints;  
9        automatically generating executable checkpoint code  
10 for execution at the checkpoints, the checkpoint code  
11 configured to store state information of the associated  
12 data objects for recovery if execution of the program is  
13 interrupted;  
14        executing the first set of object code;  
15        storing the state information in executing the  
16 checkpoint code; and  
17        upon detecting an error in execution of the first set  
18 of object code, selecting between the first set of object  
19 code and the second set of object code in resuming  
20 execution of the program.  
21

**ABSTRACT**

1  
2  
3       Method and apparatus for resuming execution of a  
4 failed computer program. A program is compiled using two  
5 compilers to generate first and second sets of object  
6 code. Checkpoints are identified in the program, and  
7 checkpoint code is generated for execution at the  
8 checkpoints. If execution of the first set of object code  
9 fails, checkpoint data is recovered and execution of the  
10 program is resumed using either the first or second set of  
11 object code. In one embodiment, the first set of object  
12 code is re-executed before trying the second set of object  
13 code.

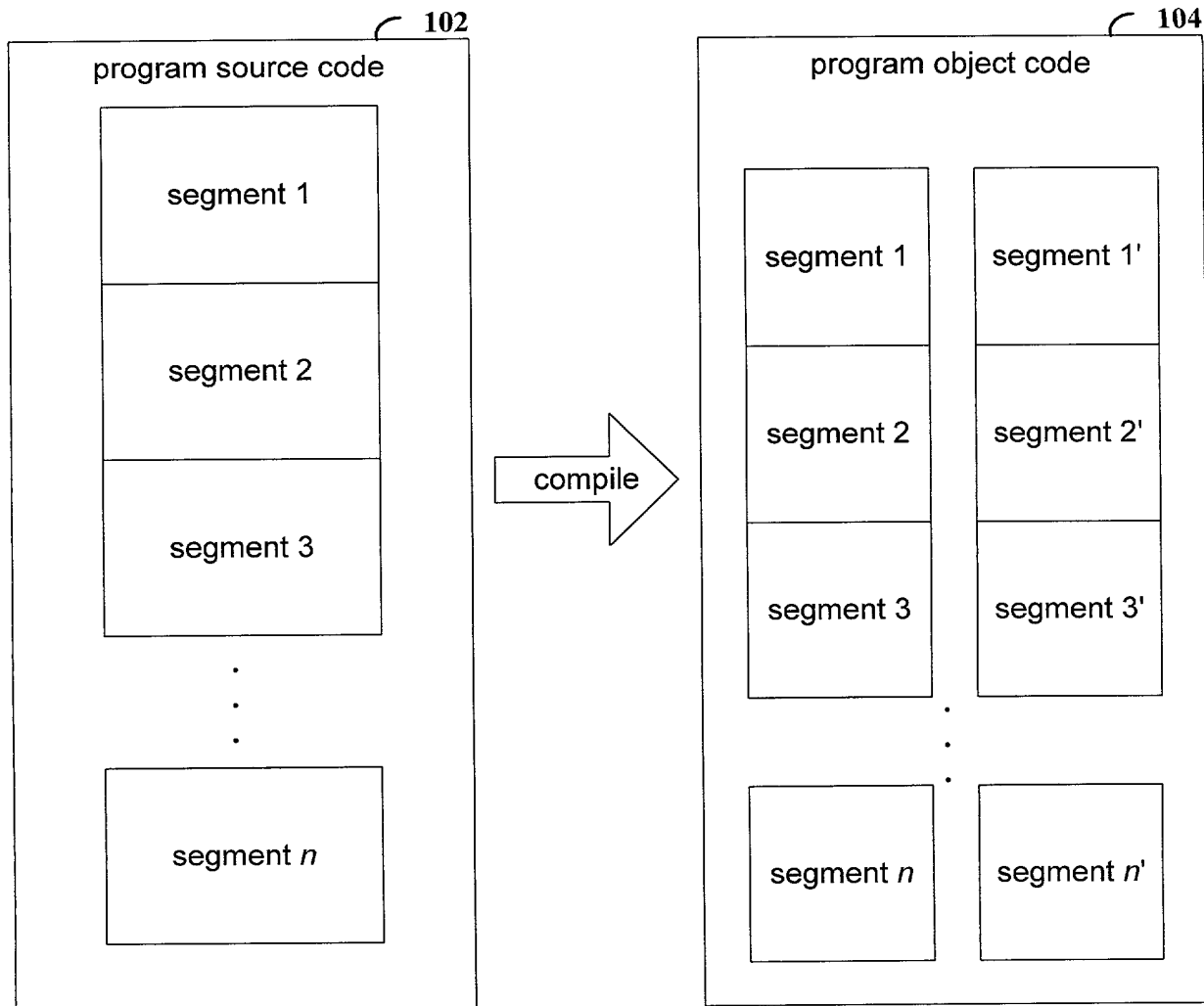


**FIG. 1**

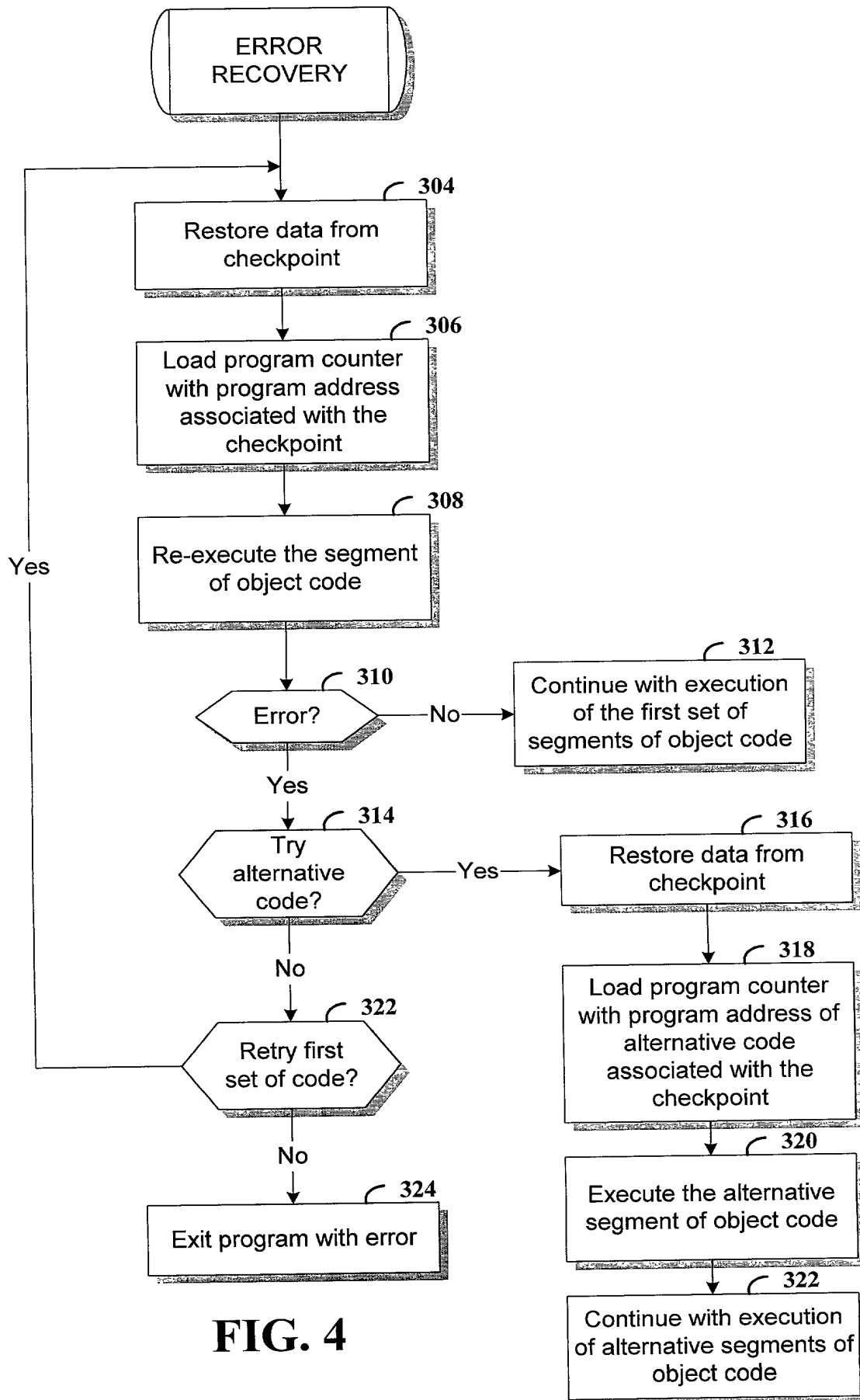


**FIG. 2**





**FIG. 3**



**FIG. 4**

**DECLARATION AND POWER OF ATTORNEY  
FOR PATENT APPLICATION**ATTORNEY DOCKET NO. 10001161-1

As a below named inventor, I hereby declare that:

My residence/post office address and citizenship are as stated below next to my name;

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled:

**Method and Apparatus for Resuming Execution of a Failed Computer Program**

the specification of which is attached hereto unless the following box is checked:

( ) was filed on \_\_\_\_\_ as US Application Serial No. or PCT International Application Number \_\_\_\_\_ and was amended on \_\_\_\_\_ (if applicable).

I hereby state that I have reviewed and understood the contents of the above-identified specification, including the claims, as amended by any amendment(s) referred to above. I acknowledge the duty to disclose all information which is material to patentability as defined in 37 CFR 1.56.

**Foreign Application(s) and/or Claim of Foreign Priority**

I hereby claim foreign priority benefits under Title 35, United States Code Section 119 of any foreign application(s) for patent or inventor(s) certificate listed below and have also identified below any foreign application for patent or inventor(s) certificate having a filing date before that of the application on which priority is claimed:

COUNTRY	APPLICATION NUMBER	DATE FILED	PRIORITY CLAIMED UNDER 35 U S C 119
			YES: _____ NO: <u>X</u>
			YES: _____ NO: <u>X</u>

**Provisional Application**

I hereby claim the benefit under Title 35, United States Code Section 119(e) of any United States provisional application(s) listed below:

APPLICATION SERIAL NUMBER	FILING DATE

**U. S. Priority Claim**

I hereby claim the benefit under Title 35, United States Code, Section 120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code Section 112, I acknowledge the duty to disclose material information as defined in Title 37, Code of Federal Regulations, Section 1.56(a) which occurred between the filing date of the prior application and the national or PCT international filing date of this application:

APPLICATION SERIAL NUMBER	FILING DATE	STATUS (patented/pending/abandoned)

**POWER OF ATTORNEY:**

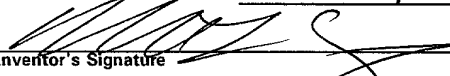
As a named inventor, I hereby appoint the following attorney(s) and/or agent(s) to prosecute this application and transact all business in the Patent and Trademark Office connected therewith:

Customer Number **022879**Place Customer  
Number Bar Code  
Label here

Send Correspondence to:  
**HEWLETT-PACKARD COMPANY**  
 Intellectual Property Administration  
 P.O. Box 272400  
 Fort Collins, Colorado 80527-2400

Direct Telephone Calls To:

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Full Name of Inventor: Michael L. Ziegler Citizenship: United StatesResidence: 1189 Audrey Avenue, Campbell, California 95008Post Office Address: 1189 Audrey Avenue, Campbell, California 95008Inventor's Signature Date 11/13/00

DECLARATION AND POWER OF ATTORNEY  
FOR PATENT APPLICATION (continued)

ATTORNEY DOCKET NO. 10001161-1

Full Name of # 2 joint inventor: Carol L. Thompson Citizenship: United States

Residence: 6937 Calabazas Creek Circle, San Jose, California 95129

Post Office Address: 6937 Calabazas Creek Circle, San Jose, California 95129

Inventor's Signature *Carol L. Thompson* Date Nov 16, 2000

Full Name of # 3 joint inventor: \_\_\_\_\_ Citizenship: \_\_\_\_\_

Residence: \_\_\_\_\_

Post Office Address: \_\_\_\_\_

Inventor's Signature \_\_\_\_\_ Date \_\_\_\_\_

Full Name of # 4 joint inventor: \_\_\_\_\_ Citizenship: \_\_\_\_\_

Residence: \_\_\_\_\_

Post Office Address: \_\_\_\_\_

Inventor's Signature \_\_\_\_\_ Date \_\_\_\_\_

Full Name of # 5 joint inventor: \_\_\_\_\_ Citizenship: \_\_\_\_\_

Residence: \_\_\_\_\_

Post Office Address: \_\_\_\_\_

Inventor's Signature \_\_\_\_\_ Date \_\_\_\_\_

Full Name of # 6 joint inventor: \_\_\_\_\_ Citizenship: \_\_\_\_\_

Residence: \_\_\_\_\_

Post Office Address: \_\_\_\_\_

Inventor's Signature \_\_\_\_\_ Date \_\_\_\_\_

Full Name of # 7 joint inventor: \_\_\_\_\_ Citizenship: \_\_\_\_\_

Residence: \_\_\_\_\_

Post Office Address: \_\_\_\_\_

Inventor's Signature \_\_\_\_\_ Date \_\_\_\_\_

Full Name of # 8 joint inventor: \_\_\_\_\_ Citizenship: \_\_\_\_\_

Residence: \_\_\_\_\_

Post Office Address: \_\_\_\_\_

Inventor's Signature \_\_\_\_\_ Date \_\_\_\_\_